**Project Scope - Deliverable 1**

**Title: Meal Recipe Database**

Group Members: Yashal Saleem, Charles Kumets, Siddharth Vadlamani, Han Nguyen, Agatha Lam, Toan Tran, Aidan Gow

**Feedback provided: Group**

We did not need to make any changes based on the feedback provided.

**Software process model: Group**

Incremental process model because we can build on to the database whenever there are more provided recipes/features.

**Software Requirements: Yashal**

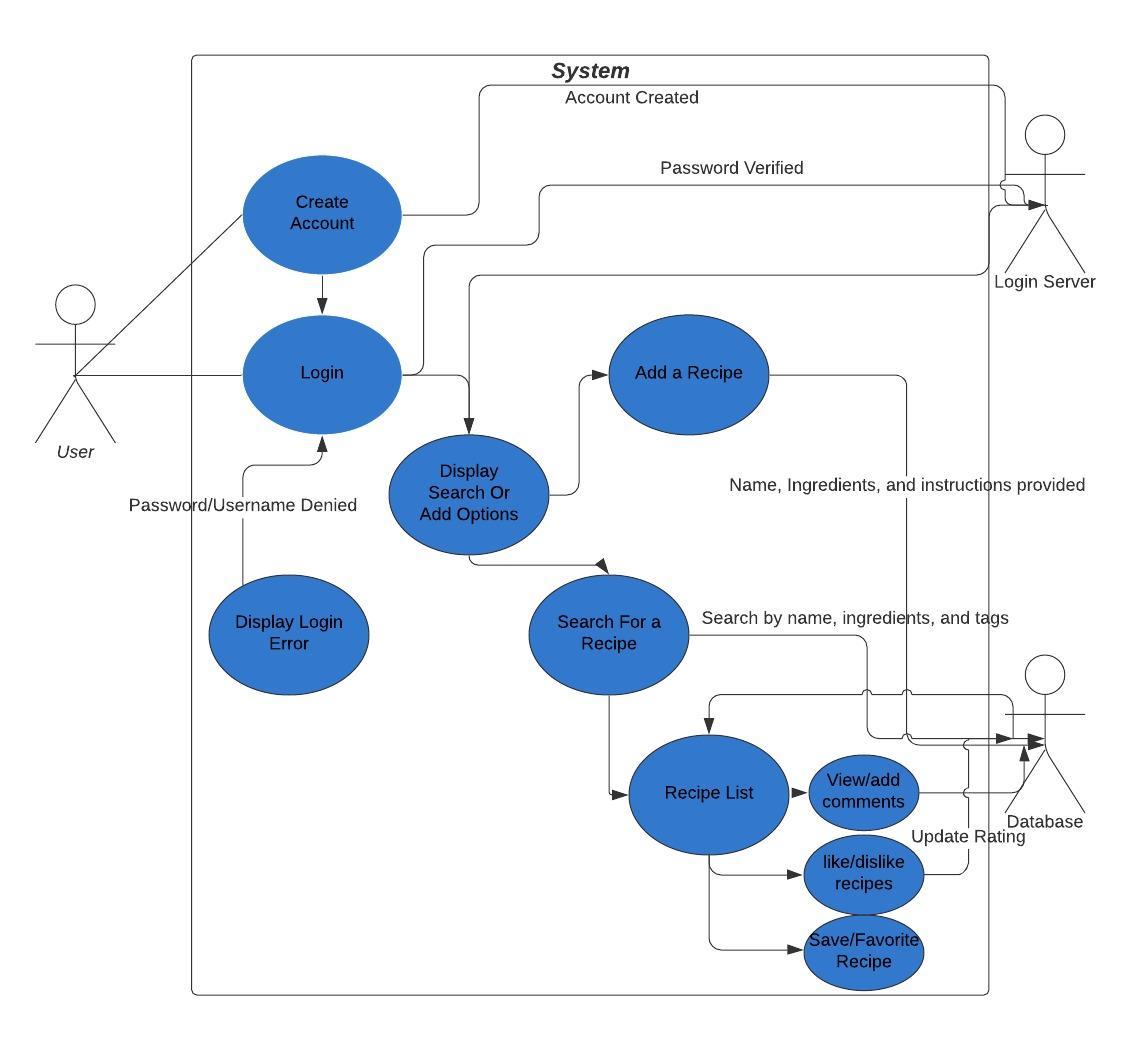
Functional requirements:

* A user can input ingredients
* A user can add new recipes as well as save/favorite recipes.
* A user will receive a recipe based on ingredients.
* A user can create an account or login.
* A user can view/add comments under specific recipes.
* A user can like/dislike recipes (rating).
* A user can search for recipes. (with tags)

Non Functional Requirements:

* Usability Requirements:
  + Should work on multiple platforms
  + Should be accessible online and somewhat offline for users.
  + Clear directions so that users can use software to full potential.
* Performance Requirements:
  + User inputs ingredients and outputs
  + Available on android/IOS
  + Programing languages include Swift(IOS), Java(android)
  + Database = Sequel (Server)
* Space Requirements
  + For the layered architecture the main recipe data will be held in one layer
  + Space for the user submission should be a set maximum they’ll be able to reach.
  + Space for storing all the submitted recipe data in general will be manipulated dynamically.
* Dependability Requirements
  + Should be able to use some functions offline to a certain extent.
  + Space for saved recipes should be large and then restrict when reaches full capacity.
  + The allocation for the recipes will be dynamic so that more can be continuously added overtime without it breaking the system.
* Environmental Requirements
  + The project will try not to push the limits of the system, most especially when it comes to storing data.
  + The different components used in the environment will be done based on how essential they are to the project. Otherwise we will end up increasing complexity and have poor performance.
* Operational Requirements
  + The database should be able to hold onto a lot of data and still be able to function at almost the same speed if there wasn’t a lot of data.
  + The time for lookup of the recipes in the database should be very low and not increase by a lot if the size of the database increases.
  + Commenting and adding recipes should be updated in a very short time when submitted, with the chance to edit what has already been posted as well.
* Development Requirements
  + The development process will consist of taking small steps with versions that will improve overtime so that it doesn’t get too overwhelmed at the beginning.
  + Testing for each feature will occur and every test case needs to be satisfied before moving onto the next step in the process.
  + Each feature and changes done in each version will be clearly documented and explained.
* Regulatory Requirements
  + The project will follow the required security regulations needed when it comes to storing user data like this.
  + Transparency in how the data will be used and held will be clear since that's an important regulation that needs to be followed.
* Ethical Requirements
  + It will be made sure that any user information that isn’t explicitly mentioned to be public will not be shared.
  + It will be made explicitly clear to the users what information they post can be used for. This is for both in-app usage and out of app usage of the data.
* Accounting Requirements
  + Money and resources put into this project and divided into it will be clearly outlined and transparent.
  + Profit from this project will first take into the account of reinvesting some of it to further improve the project.
* Safety/Security Requirements
  + Sensitive user data will be encrypted.
  + Precautions will be done to try to limit how much sensitive information a user may unknowingly leak to other people.
  + Ways to reset password and provide backup ways to login if the user loses access to the account via the normal method.

**Use Case Diagram: Aidan**



**Architectural Design: Aggie**

****

**Sequence Diagram: Toan/Charles**



**Class Diagram: Siddharth/Han**

